# SECUENCIADOR PROGRAMABLE DE 8 CANALES

Debe haber alguna cuestión sicológica que explique el por que de la atracción que tiene sobre los humanos las lucecitas de colores (¡un psicólogo ahí!), mas que nada las que encienden y apagan (si es con algún patrón, mejor). No pretendemos desde estas paginas explicar el fenómeno, pero si podemos utilizar un microcontrolador para sacar provecho a este fenómeno, y diseñar un circuito que permita generar patrones de luz a través de LEDs o incluso mediante lámparas incandescentes (las lamparitas comunes) de 110V o 220V, hasta unos 800 Watts por canal.

Este proyecto que voy a explicar nos permitirá elegir uno de 16 programas posibles, mediante dip-switches colocados en la plaqueta del microcontrolador, cada uno de ellos ejecutara una secuencia de luces diferentes. Las luces en cuestión pueden ser simples LEDs, ideales para moddear nuestro ordenador, o bien, mediante una etapa de potencia que también describiremos, manejar cargas mas grandes, como luces o lo que se nos ocurra, siempre que no superen el máximo permitido.

### > PIC16F628A

Este es el nombre del microcontrolador que usaremos en este proyecto. Si bien tiene 18 pines igual que nuestro viejo y venerado 16F84A, tiene varias ventajas respecto de este. Se trata de un PIC bastante mas nuevo, e incluso hasta mas económico que el 16F84A. Entre las ventajas que tiene (son demasiadas como para enumerarlas a todas) quizás las mas importantes son las referidas al tamaño de la memoria disponible para el programa, que es de 2 Kb (dos veces la del 16F84A), la posibilidad de utilizar un oscilador interno, con lo que nos ahorramos el cristal y los dos capacitares, además de poder utilizar esos dos pines como E/S, la inclusión de dos comparadores de tensión en el PORTA, etc.

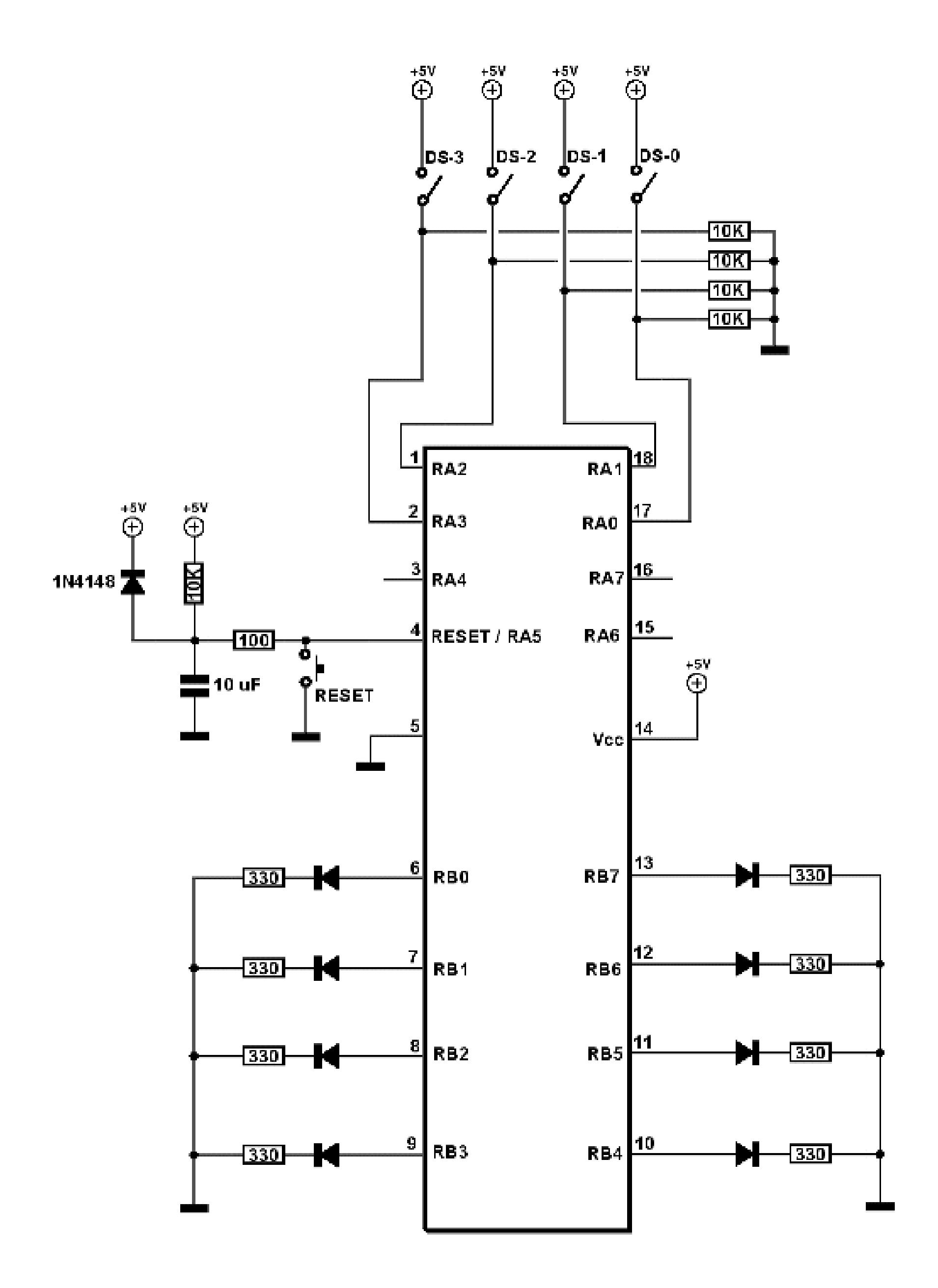
Como ventaja, desde el punto de vista del programador en ASM, utiliza el mismo set de 35 instrucciones que se utiliza en el 16F84A. Si. Como haremos nosotros, utilizamos un lenguaje de alto nivel para programarlo, no notaremos ninguna diferencia.

Como siempre, recomendamos enfáticamente la lectura de la hoja de datos de cada microcontrolador que usemos, para extraer toda su funcionalidad, y no cometer errores en su utilización. Piensen que estas “hojas de datos” son casi un libro, ya que frecuentemente tienen algunos cientos de páginas.

### > El circuito

El circuito eléctrico, que podemos ver en la figura, no podría ser más simple. Esa es la ventaja de utilizar microntroladores, donde la lógica del circuito se decide mediante el software y no mediante el hardware. Además de la sencillez, el hecho de tener los diferentes esquemas de encendido de las luces codificadas en un programa es que si en algún momento tenemos necesidad de cambiar las secuencias por algún motivo, simplemente reprogramando el PIC accederemos a nuevas prestaciones, sin necesidad de enchufar el soldador.

---



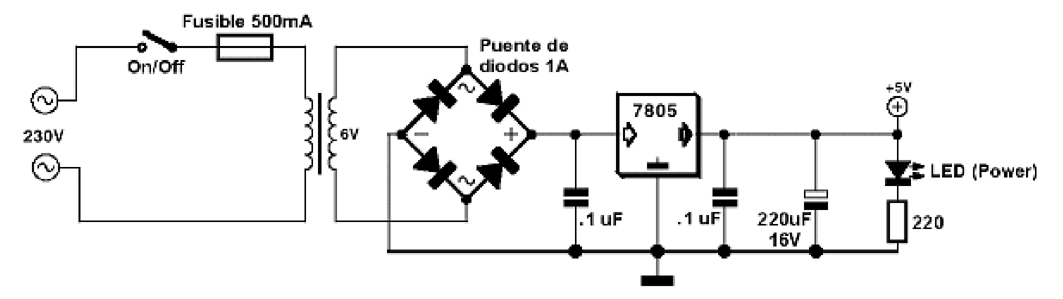

Como podemos apreciar, hay 4 llaves conectadas a los pines 18, 17, 1 y 2, que corresponden a los bits 0, 1,2 y 3 del PORTA del 16F628A. Estas llaves pueden ser dip-switches montados en la propia placa, o si queremos una forma mas fácil de cambiar de programa, podemos poner 4 llaves en el gabinete que elijamos para albergar el montaje y cablearlas hasta los agujeros del circuito impreso. Existen 16 combinaciones posibles para estas llaves (2^4 = 16), de ahí el numero de programas posibles.

Las resistencias que están entre cada uno de estos pines y el pulsador cumplen la función de poner a masa las entradas cuando el pulsador esta abierto. Si no hacemos esto, las entradas se comportan como una especie de antena, y pueden aparecer aleatoriamente "0" y "1" en ellas, provocando el comportamiento errático del programa.

Todo el PORTB del PIC (pines 6, 7, 8, 9, 10, 11,12 y 13) se conectan a los LEDs, o en caso de querer utilizar el modulo de potencia, deberemos unir con un cable cada uno de estos pines a las entradas de dicho modulo.

Por ultimo, se dispone de un botón de reset, para reiniciar el programa, y una etapa de alimentación construida alrededor de un transformador, un regulador de voltaje LM7805 y algunos capacitores para filtrar algún eventual "ripple" residual que pueda tener la fuente de alimentación que usemos. También se incluyo un LED para indicar que la placa esta alimentada.

---




Esta es la fuente de alimentación sugerida para el montaje.

## > El software

El software que va a manejar nuestro secuenciador es bastante sencillo de desarrollar, solo debemos configurar correctamente los puertos del PIC, y a continuación definir el estado de cada una de las salidas dependiendo de la posición de los interruptores de entrada.

Solamente vamos a ver el "encabezado" del programa, es decir, la sección donde se hacen las declaraciones de las variables, configuración de los puertos, etc. y un solo efecto de los 16 que se pueden programar.

Vamos a suponer que queremos crear un efecto, que se va a mostrar cuado los 4 interruptores de entrada estén abiertos (seria la combinación "0000"), consistente en una luz que se desplaza de izquierda a derecha, y luego al revés. Los mas memoriosos (decir "viejos" me pareció inadecuado) recordaran una serie llamada "El auto fantástico" donde el vehiculo protagonista tenia en el frente una luz de este tipo. Yo nunca la vi, pero me contaron…. :)

```
program Secuenciador

main:       'Este es el cuerpo del programa

    'Declaracion de Variables
     dim i as byte
     dim p as byte

    PORTA = 0
    adcon1 = 6     'Configuro puerto A como Digital I/O

    TRISA = %00001111    'Puerto A0,A1,A2,A3 como entradas, resto salidas
    TRISB = %00000000    ' PuertoB todo como salidas
    PORTB = %00000000    ' Apago los LEDs conectados al puertoB

    while TRUE           ' Comienzo un bucle infinito
        'Calculo el numero de programa seleccionado con los switches
      P = PORTA.0 + PORTA.1*2 + PORTA.2*4 + PORTA.3*8

      if P = 0 then   ' Programa "0000"
          ' Enciendo secuencialmente las luces,
          ' con una demora de 200 milisegundos entre una y otra
          PORTB = %00000001   ' Enciendo el LED 0
          delay_ms(200)       ' Espero 200 milisegundos

          PORTB = %00000010   ' Enciendo el LED 1
          delay_ms(200)       ' Espero 200 milisegundos

          PORTB = %00000100   ' Enciendo el LED 2
          delay_ms(200)       ' Espero 200 milisegundos

          PORTB = %00001000   ' Enciendo el LED 3
          delay_ms(200)       ' Espero 200 milisegundos

          PORTB = %00010000   ' Enciendo el LED 4
```

```
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00100000    ' Enciendo el LED 5
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %01000000    ' Enciendo el LED 6
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %10000000    ' Enciendo el LED 7
        delay_ms(200)        ' Espero 200 milisegundos

        'Ahora, programo el "regreso" de la luz….
        PORTB = %01000000    ' Enciendo el LED 6
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00100000    ' Enciendo el LED 5
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00010000    ' Enciendo el LED 4
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00001000    ' Enciendo el LED 3
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00000100    ' Enciendo el LED 2
        delay_ms(200)        ' Espero 200 milisegundos

        PORTB = %00000010    ' Enciendo el LED 1
        delay_ms(200)        ' Espero 200 milisegundos


    End if


    ' Aquí irian las demas preguntas para ver si
    ' el programa seleccionado es otro (hay que comprobar
    ' desde "0000" a "1111" (de 0 a 15)

    if P = 1 then   ' Programa "0001"
         ' Programa 1………
   End if

   ' etc,etc,etc…..


  wend                ' repetir el bucle.

end.   'Fin del programa
```

Si compilamos el programa, utilizando Mikrobasic como compilador, veremos que cuando tenemos todos los interruptores abiertos, los LEDs se desplazan tal lo programado, pero en cualquier otra combinación los LEDs permanecen apagados. Esto se debe a que en el listado de ejemplo no hemos definido acción alguna para las demás configuraciones de los switches. La idea es que cada uno programe los efectos que desee.

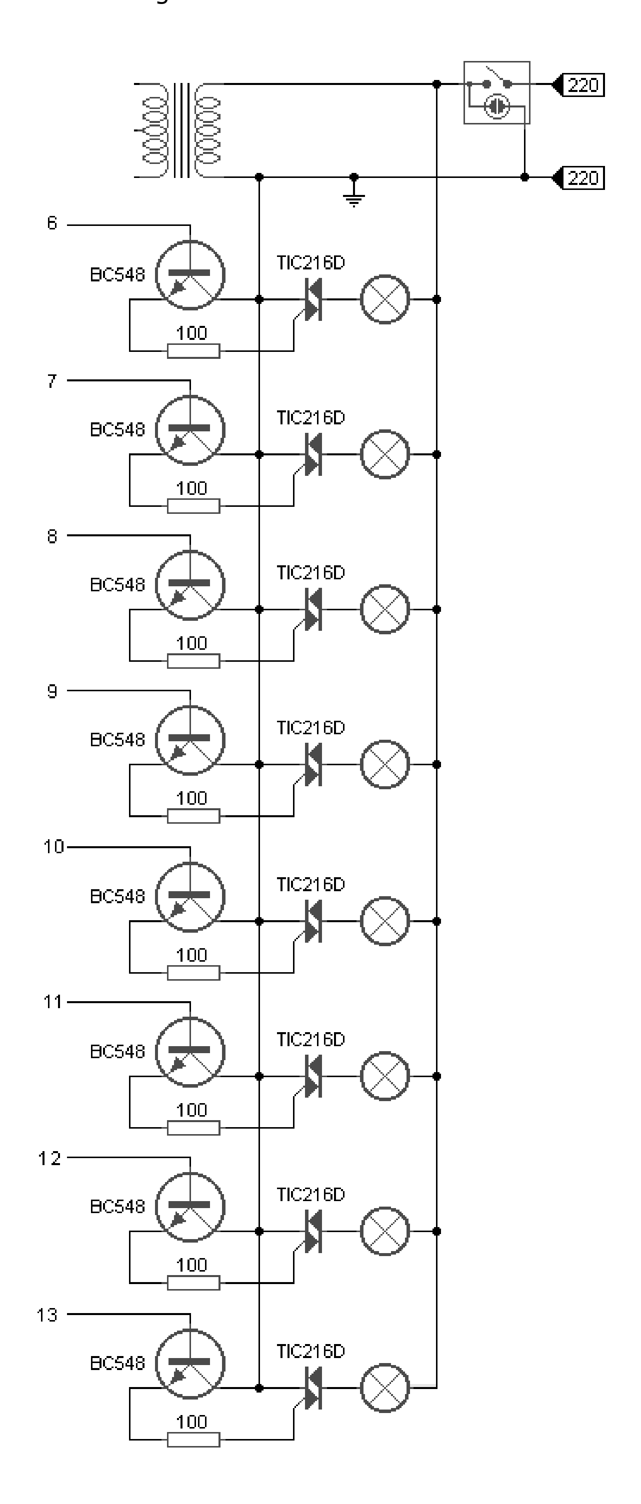
Por supuesto, este programa podría haber sido mucho mas eficiente (desde el punto de vista de la memoria requerida) o elegante, pero preferí la claridad antes que lograr un código compacto. Como siempre, intentamos dar solo una guía como para que los lectores desarrollen sus propias rutinas, y por supuesto los alentamos a que modifiquen TODO.

Aquellos que están siguiendo la guía del PIC SIMULATOR IDE podrán, de forma muy fácil, convertir el programa sugerido a ese formato.

### > Usando lámparas

Hace algún tiempo, recibí mails de algunos lectores que me pedían algún circuito para controlar luces que usaban en fiestas, etc. Me pareció apropiado dotar a este proyecto de una etapa de potencia, para que el lugar de encender pequeños LEDs podamos manejar lámparas de hasta 800W en cada canal.

Se deben construir 8 etapas iguales, tal como muestra la figura, una por cada canal, y conectarlas en el lugar de los LEDs. En el esquema figuran a que pines del PIC hay que conectar cada uno de los transistores BC548B que manejan los triacs. Todo lo demás es igual.




Si la(s) lámpara(s) que vamos a conectar en cada canal no consumen mas de 100W, podemos usar los triacs sin disipadores de temperatura, pero si queremos agregar consumos mas importantes, inevitablemente deberemos dotar a cada uno de los triacs con un disipador adecuado, si no se destruirán.

Es MUY importante recordar que cuando trabajamos con la tensión de red (110V o 220V) cualquier descuido puede terminar muy mal, así que como siempre recomendamos encarecidamente NO encarar esta parte del proyecto si no tienen los conocimientos suficientes para manejar tensiones elevadas.

Esta etapa deberá estar contenida en un gabinete, preferentemente plástico o muy bien aislado, con salidas para los cables únicamente, para evitar tocar sin querer alguna parte del circuito que este con 110V/220V.